PROGRAM PROTECTION NEWSLETTER
JULY 1985

WHAT IS A PROGRAM PROTECTION NEWSLETTER?

It is a monthly newsletter that will deal with program protection schemes of all types. Each month will cover 3 to 5 specific examples of different program protection schemes. Some will be very easy schemes, others may be very difficult. Some issues will cover word processors, data bases, spread sheets and games. Many different programs will be selected and each will be fully explained as to their type of protection.

Each program example used will be for illustrative purposes only. It is not the intention of this newsletter to promote or encourage illegal or unauthorized duplication of any copyrighted program. UNAUTHORIZED DUPLICATION OF A COPYRIGHTED PROGRAM IS ILLEGAL!!

WHY WOULD ANYBODY NEED A PROGRAM PROTECTION NEWSLETTER?

Many programs have a very sophisticated protection scheme and can not be used on some disk drives (MSD, 4040, 8050 etc.). Other programs use errors on the disk that can be very harmful to the disk drive. Many other people just prefer to use one disk for all their programs of one specific type (i.e. utilities).

According to federal copyright law you are allowed to make an ARCHIVAL copy of any program that you have purchased. You are also allowed to modify your program as long as the modifications are essential to the program's operation. Some protection schemes do not allow the program to be directly backed up. They may require the use of sophisticated copy programs or extensive knowledge of MACHINE LANGUAGE to understand. In either case the average user may not be able to obtain an archival copy of their valued programs. Another group of people just want to know more about their computer and its associated components. They want to know how programs are protected and how their machine operates. They want to know how to modify their own disks and how to protect their own programs.

Whatever reason you have for wanting to learn about program protection, this newsletter is for you!

PASS THIS COUPON ON TO YOUR FRIENDS WHO WISH TO LEARN MORE ABOUT PROGRAM PROTECTION SCHEMES. **BETTER YET, USE IT TO RENEW YOUR SUBSCRIPTION FOR THE NEXT YEAR, AUG. 85 TO JULY 86.**

SUBSCRIPTION PRICE $35.00 POST PAID IN U.S. AND CANADA
($45.00 FOR 1ST CLASS FOREIGN MAIL DELIVERY)

NAME ______________________________

ADDRESS ______________________________

CITY,ST,ZIP ______________________________

Renew by August for
continuous delivery.
See the May issue for
special renewal offer.

CSM SOFTWARE INC.
P. O. BOX 563
CROWN POINT, IN 46307
(219) 663-4335

Well, this issue marks the end of the first year of the PROGRAM PROTECTION NEWSLETTER, and the end of your current subscription. Looking over the past 12 issues, we've covered a lot of territory. We've examined a wide range of protection schemes, from simple bad block methods (numerous examples) to custom DOS routines (see Creative Filer in this issue). We've explained different encryption methods (F-15 Strike Eagle, 6/85 and others), how to tell if a routine is being left in the disk drive (UNGUARD, 9/84), how to get code from under the I/O devices at $D000 (The Music Processor, 5/85) and how to find entry points into routines (numerous examples). We've covered cartridges extensively, explaining how the EXROM and GAME lines control the memory configuration (Advanced cartridges, 8/84), how to locate code that tries to write to the cartridge (Hesmon, 10/84 and MBA-TOR, this issue), how to make a disk copy of a cartridge (numerous examples) and even how to make a cartridge copy of a disk program (Easy Script, 1/85). We've also introduced you to the Promenade EPROM programmer and its many uses.

We've given you valuable technical information, to enable you to do many things that are supposed to be for 'experts' only. We showed you how to modify the DOS to read and write 40 tracks (12/84). We showed you how to work 'inside' the drive, including how to execute DOS routines directly, how to interpret error codes, change densities and move the head directly (2/85). We've detailed the hardware limitations of the 1541 (4/85) and a possible way to get around one of them (writing adjacent half-tracks, 6/85). We've looked into the KERNAL numerous times, including the differences between the various ROM revisions (5/85) and how to make your own modifications to the KERNAL (6/85).

With our product reviews we've brought new products to your attention, both to recommend the good ones and steer you away from the bad ones. If we kept you from wasting your money on one of these 'duds', the newsletter has paid for itself. With a slew of new 'miracle' products coming onto the market soon (with price tags to match), you'll find this kind of dependable product evaluation is even more important. Last but not least, we've shown you how to make archival backups of over 50 programs. If we've enabled you to back up just ONE of your original programs, we've saved you the cost of your subscription (not to mention saving your disk drive from an early grave). Since copy protection is becoming more sophisticated almost daily, you'll need our kind of expertise even more in the next year. For those of you who are working to protect your own software, the newsletter offers a unique opportunity to see what others in the industry are doing, what works and what doesn't.

If all this sounds like we are tooting our own horn, well, we are. We're proud of the service we provide, and we spend a lot of time and energy to make sure it's up to our standards, because we believe in doing it right or not at all. We also believe that we've succeeded in doing it right, and we hope you feel the same way. If not, please write to us and let us know. We need your feedback and your support. So take a minute to recall all that you've learned from the newsletter in the last year, and ask yourself - wasn't it more than worth the price? Then mail in your subscription, and tell all your friends that have been reading your copy of the newsletter to do the same. See you next month.

The program is MBA-TOR(tm) copyright 1984 by AEA (version 27.DEC.84)

TYPE OF PROTECTION: This is a 16K cartridge program that uses indirect, indexed addressing - STA (88),Y. This form of addressing uses a pointer on zero page, and the value contained in the Y-register, to calculate the memory location to be acted upon. This cartridge is an AUTO-START cartridge, but it does not use the common CBM80 as found in most other cartridges that reside at $8000-$BFFF. This auto-start feature will be looked at in detail in just a few minutes.

HOW TO BACK UP: Although other versions of this program may be directly backed up to disk with the CARTRIDGE BACKER, this version of the program may not be backed up. A copy of the cartridge may be made by burning EPROMS and installing them on a cartridge board.

HOW TO MAKE A WORKING COPY: The easiest way to locate the program's protection is to use the COMPARE AND REPAIR function of the CARTRIDGE BACKER program. This function allows the user to narrow down the possible locations for the programs protection. It is possible to do this program without the use of the CARTRIDGE BACKER, so we will include both methods.

1. First we need a disk (RAM) version of the program. Use either OPTION 1 or 2 from the CARTRIDGE BACKER (CB) program (both will be identical); just be sure to obtain a printed report. For the sake of clarity, name your object file 'MB'. If you don't have the CB software, use the techniques outlined in the Program Protection Manual (I or II). For review we will cover the technique here.

    a. Insert your cartridge and de-activate it (turn off the cartridge switch or turn off the proper switches on your expander board).

    b. Turn on your computer, load HIMON and then SYS49152.

    c. Activate your cartridge (by the switch or via your expander board).

    d. Save out the cartridge memory with the following command:
    S "MB.OBJ",08,8000,C000

2. Now it is necessary to execute the RAM version of the program. If you are using the CB, simply load and execute the auto boot and allow the program to crash. Flip switch 2 ON and RESET the computer. Now flip switch 2 OFF and load the CB software. Follow the steps in OPTION 5 (COMPARE AND REPAIR) to identify the location that has been written to ($85AF). Note that location $8000 will be changed by the RESET routine of the computer and NOT by the program itself.

    If you don't have the CB, follow these instructions:

    a. After you save the program out to disk (and you're still in the monitor), transfer the program into the RAM under the cartridge:
    T 8000 BFFF 8000

    b. Now de-activate the cartridge via the switch(es). Turn off the BASIC ROM by changing location $0001 from $37 to $36. Use the following command to execute the RAM copy: G 805A

    The program will crash. RESET your computer and re-enter your ML monitor.

c. Turn off the BASIC ROM (change location $0001 to $36) and transfer the code down to $2000:

```
T 8000 BFFF 2000
```

d. Load the disk version of this program:

```
L "MB.OBJ",08
```

e. Compare the two versions of the program:

```
C 8000 BFFF 2000
```

f. The comparision will result in finding locations $A000 and $85AF as being different. In this case, location $A000 was changed by the RESET routine and $85AF was written to by program.

3. Now that we have established that location $85AF is where the cartridge is writing to itself (thereby causing the program to crash), we can hunt for the code that writes to $85AF. By the way, location $85AF gets changed to the value of $00. Since CB does not fix this program on its own, you can be reasonably sure that protection will be indirect indexing (STA (XX),Y). First thing to try is hunt for where the program sets up the pointers to do the indirect indexing. We can usually find this by doing a simple hunt for LDA's (A9) with the individual address bytes. Try the following hunts:

```
H 8000 BFFF A9 AF            and
H 8000 BFFF A9 85
```

The following locations will be returned: 80DE & 80E5. Take a look the programs ML code from $80DE on. It will become apparent, during investigation, that this area of code (from $80DE on) is setting up pointers at $88 and $89 (among other things). The pointers ($88 & $89) are pointing to location $85AF. At $8112 the program loads the accumulator with the value of $00 (LDA #$00). At $811A the program transfers the value in 'A' ($00) to the 'Y' (TAY), thereby also setting the 'Y' to $00. Next (at $811B & 811C) the program stores the value in the 'A' at location $85AF with the command STA ($88),Y. Remember that locations $88 & $89 point to $85AF and that the value of 'Y' is $00, so the final location will simply be $85AF.

Keep in mind that a cartridge is ROM based and will not be changed if it tries to write to itself, whereas the same program residing in RAM can write to itself and cause the program to crash. All that is required to prevent the program from writing to itself is to change the STA ($88),Y to NOP's. This is accomplished by changing the code at $811B & $811C to EA's. Just by changing these two bytes to NOP's the program will now run from RAM. If you had the CB software and printed a report of the possible locations where program protection may be hidden, you would find that the very first INDIR type of addressing listed is where the protection was located. This is why the CB software is so very valuable. For the few cartridges that CB will not backup, it is possible to use the report generated to find the protection.

4. If you are using the CB software all you need to do is save the modified version back to disk. You should replace the original version with the modified version:

```
S "@0:MB.OBJ",08,8000,C000
```

At this point the CB version of this program will now load and execute properly from disk. It will require a little more work for those of you that don't have the CB.

If you don't have the CB it will be necessary to add a little bit of additional code to the program. This extra code will turn off the BASIC interpreter and then jump to the proper entry point of the program. For the sake of convience we will add this code beginning at $C000 (49152). This way when you load the program back in all that is required is to SYS49152 to execute this program. Use the 'M' command of your ML monitor to enter the code:

```
C000 A9 36 8D 01 00 6C 00 A0
```

```
C000 A9 36    LDA #$36     (DISASSEMBLY OF CODE)
C002 8D 01 00 STA $0001    TURN OFF BASIC
C005 6C 00 A0 JMP ($A000) INDIRECT JUMP TO START OF PROGRAM
```

Save the modified version of this program back out to disk with the following command:

```
S "@0:MB.OBJ",08,8000,C008
```

To execute this program just LOAD "MB.OBJ",8,1 then SYS49152.

5. YOU'RE DONE

A final comment as to how this program auto-starts without the CBM80. As you recall this is a 16K cartridge that resides from $8000 to $BFFF. The cartridge will turn off the BASIC interpreter ROM and force the microprocessor to look to the cartridge port for the entire block of memory from $8000 to $BFFF.

During the normal initialization routine of the C-64 the computer will look for a CBM80 at $8000. If it is found the entire initialization process will be turned over the cartridge program. It must be pointed out that the initializattion process is a quite lengthy and complicated process. If the CBM80 is NOT found the normal initialization will be continued by the computer. At the end of normal initialization the computer will perform an indirect jump into the BASIC interpreter - JMP ($A000). This jump uses memory locations $A000 and A001 as vectors. Normally these locations point to the REAL start of BASIC. When we install THIS cartridge the vectors at $A000 have been changed to where the program begins. This is an easy way to let the computer perform all the initialization routines and leaves the programmer free to concentrate on writing his program (and not on initializing the computer).

The program is CREATIVE FILER (tm), Copyright 1984 by Creative Software.

There are at least two versions of the protection scheme for this program. The newer version has two files called LOADER and X1. If yours has two files called CF and TI, you have the older version. We'll cover both of them, starting with the newer, more interesting one. In fact, the new version is such a good learning tool that even if you don't like the database program itself (a distinct possibility), you should buy the disk for the protection scheme.

NEW VERSION:

TYPE OF PROTECTION: The original disk contains special information written with altered density on track 36. Also, the protection code is encrypted.

HOW TO COPY: Some of the new nibble copy programs that can handle altered bit density and extra tracks will work.

HOW TO MAKE A WORKING COPY WITHOUT ERRORS ON THE DISK:

1. Format a new disk with the name "CREATIVE FILER" (any ID will do).
2. Load a track/sector editor and run it.
3. Read in block 18/0 from the NEW disk. Change byte $50 from $13 to $12 and byte $52 from $FF to $F7. Write the block back out to 18/0. (Note: bytes are numbered starting at $00, not $01)
4. Read in block 20/11 from the NEW disk. Change bytes $00-05 to $A9 FF 8D FF 01 60. Write the block back out to 20/11.
5. File-copy all the files from the original disk to the new copy.
6. That's it - you're done.

This is practically a 'textbook' protection scheme, well worth examining in depth. There is nothing startlingly new about it, but it does contain an effective custom DOS routine combined with good use of encryption. If you have PPM Vol. II, you'll find similar routines covered there. The protection code is contained in three places: in the file "X1", in block 20/11 and on track 36. The boot program "LOADER" loads and executes the main protection routine "X1" at $7000-74A9. "X1" starts by altering the character set, setting some sprite data and printing the title screen. Then it adds up a section of itself ($7278-BE) using ADC (ADD WITH CARRY). The resulting one-byte checksum ($A7) is then EOR'ed (exclusive-or'ed) with most of the rest of the code ($72BF-74A7) in order to decrypt it. Actually, only part of the code is fully decrypted at this point; the rest will be EOR'ed again later.

Next, a major subroutine at $731C is called. It starts by opening up a random access file (#) and reading in block 18/0, through a U1 command at $7489-9A. This block contains the BAM and disk name. Then the routine sets the buffer pointer to the start of the disk name, using a buffer-pointer (B-P) command at $749B-A6. It reads the disk name into $6002-0F and then compares it to the name "CREATIVE FILER", prestored at $7004-11. If there is any mismatch, the routine crashes. The routine uses the KERNAL routines LISTEN ($FFB1), CIOUT ($FFA8) and UNLISTEN ($FFAE) to send the "U1" and "B-P" commands. This can be a form of protection since these routines may not be as familiar as the normal CHROUT ($FFD2) routine.

Having passed the first protection check, it really gets down to business. It opens up another file to the drive, this time specifying buffer #1 ($0400-FF). As explained last month in the article EXECUTING 1541 ROUTINES, some custom DOS routines must be loaded into a particular place in memory, and this can be done by specifying the buffer # rather than letting the drive pick one for you. In this case, a DOS routine is loaded from block 20/11 into buffer #1 and executed, all in one step. This is done with a handy block-execute (B-E) command at $7054-66. Like many custom DOS routines, you don't have to know what this routine is actually doing, just what value is returned eventually. This DOS routine is interesting enough that we'll take a look at it anyway. Remember, this routine is executed in drive memory.

The first thing the custom DOS routine does is disable interrupts (SEI) so it has total control of the drive. Next, a small routine decrypts the rest of the code ($0412-F1) by EOR'ing it with the value $AC. Then the decrypted code is executed. It initializes the R/W electronics (JSR $0461) and sets up two track pointers. These pointers specify that the head is currently on track 20 (from the B-E command) and that it would like to go to track 36 next. A subroutine at $0471 moves the head based on these pointers and waits a bit for the head to settle in. This subroutine also sets the proper bit density (clock speed) for the track being moved to. This is a very handy routine! After returning to the main DOS routine, the density is reset to the value for track 26. The head is now on track 36 ready to read some information at track 26 density.

The program code at $0432-3E waits for a sync mark to come along (using a subroutine at $04E6-F1) and then waits for a special marker byte, $5C. After the $5C is detected, the routine reads the next 256 bytes into drive memory at $0500-FF, and then 69 more bytes into the OVERFLOW BUFFER at $01BB-FF. A subroutine in the normal DOS ROM at $F8E0 is called next. Recall that raw data from the disk is in a special code called **GCR**, not in hex. The routine at $F8E0 is normally used to process a header block from disk. It converts the bytes in the OVERFLOW BUFFER from GCR to HEX, saves the first hex byte (header block id) separately and puts the rest of the hex into a regular buffer, in this case at $0500. After this subroutine finishes, the new code at $0500 is executed with a straight JMP. This code leaves an $FF byte at location $01FF in drive memory and terminates the block-execute command (finally!). The disk protection has now been checked - the $FF value at $01FF is the key to the protection scheme.

Meanwhile, back at the computer... after issuing the B-E command, the code in the computer sends a memory-read (M-R) command to read the value left at $01FF in drive memory. The value received is stored at $70A8 in the computer. If this value isn't equal to $FF, the program crashes. Otherwise we're over the hump, and it's all downhill from here (with a few detours). The next part of the routine adds up the section of code from $731C-74A4 (using ADC) to get a one-byte sum ($8D). This value is used to EOR two sections of code, $72BF-744B and 74A5-A7. This is interesting because all of this memory was already EOR'ed earlier. The net effect is to finally decrypt the areas $72BF-731B and $74A5-A7, and RE-ENCRYPT the rest of it! At this point, we have come to the end of the major subroutine. We return back to the main routine at $72BF, an area which was just decrypted. This last section of code looks a little involved, but it simply loads in the files "CODE", "MATH", "BCDF" and "GOGO" at their normal locations. Finally, all open files are closed and the two bytes at $8000-01 (cartridge cold start vector) are used as an indirect vector to start the main program.

To summarize the protection method, the disk name is checked and a custom DOS routine is executed which leaves a key value in drive memory. We take care of the disk name in step 1 of our directions for removing the protection. The rest of the steps are concerned with the B-E routine. In step 2, we modify the BAM to mark block 20/11 as used, so the regular files won't wipe it out. Then in step 4, we put a short routine into 20/11 which simply stores our key value into drive memory and returns. The disassembly of this routine is:

```
0400  A9 FF      LDA #$FF
0402  8D FF 01   STA $01FF
0405  60         RTS
```

This routine replaces the custom DOS routine, and thus makes the special data on track 36 unnecessary. After copying the regular files you are all set. Other files can be added to this disk safely, as long as you don't validate the disk.

OLD VERSION:

TYPE OF PROTECTION: The program checks track 2, sector 0 for an error (any). The disk name is also checked.

HOW TO COPY: Use any copy program which can copy errors, such as Omniclone.

HOW TO MAKE A WORKING COPY WITHOUT ERRORS ON THE DISK:

1) Copy the disk with any copy program which doesn't put errors on the backup disk. You may file copy the disk if you prefer.
2) Load and run HIMON. Fill memory with F 0801 7FFF 99.
3) Load the protection routine with L "TI",08
4) Display memory with M 707D. Change bytes $707D-7F from $20 CF FF to $BD 54 71 .
5) Display memory with M 7127. Change $7127 from $32 to $30.
6) Replace the altered routine with S "@0:TI",08,7000,741E
7) That's it - you're done!

This program first reads the disk name and compares it against a prestored value at $7154. The patch in step 3 above alters the routine so that it compares the prestored value with itself. The program also checks block 2/0 for any error, using the time-honored CMP #$32. Step 4 changes it to CMP #$30 as usual so that it passes on an error-free disk. The disk can now be file copied or have other files added to it.

CORRECTION - CUSTOMIZING THE KERNAL

There was an error in the article last month about CUSTOMIZING THE KERNAL. In the section on reprogramming the SHIFT-RUN/STOP combination, the location of the new command **MINUS 1** should be put at $E5F4-5. That is, set $E5F4-5 to **$BF E4** rather than $C0 E4 as shown. Thanks to the Unknown Reader who brought this to our attention (and whose name unfortunately escaped me).

The program is Copy-Q II, Copyright 1985 by Q-R&D.

TYPE OF PROTECTION: This program uses different types of non-standard formatting as well as half-tracking.

HOW TO COPY: This disk is very difficult to copy; we could not find a copy program which would work consistently.

HOW TO MAKE A WORKING COPY WITHOUT ANY ERRORS ON THE DISK:

1. The original disk contains 'templates' which are used to copy different programs. The 'templates' are not in files, but are stored on tracks 1-17 in various non-standard ways such as on half-tracks. The first thing to do is make a copy of the original disk, using Copy-Q II to copy itself. This copy will not work, but the 'templates' will be copied correctly. We'll modify the Copy-Q II program and put it on the copy disk with the templates.
2. We need to mark the first 17 tracks 'used' in the BAM so when we add the Copy-Q II program it will not use these tracks. Load a track and sector editor and run it. Read in block 18/0 from the COPY disk. Change bytes $04-47 to all $00's. If your T/S editor has an ML monitor, you may prefer to use its 'fill' command to change these bytes. Write the block back out to disk. Exit the T/S editor back to normal BASIC. (Note: bytes are numbered starting at $00, not $01)
3. Load a monitor at $8000 or higher, such as HIMON, but do not execute it. Type 'NEW'.
4. Insert the ORIGINAL disk and type LOAD "MENU",8 After the menu loads, type POKE 2942,96. This puts an RTS at $0B73 in place of JMP $0900.
5. Now RUN the menu program. The main program will load in about 30 seconds, but the RTS at $0B73 will cause it to crash. You'll still be able to enter commands, however.
6. Execute your monitor, which should still be intact. The Copy-Q II program is now in memory at $0800-3FFF. We'll preserve a small section by transferring it up in memory: T 0800 080F 4010
7. Next, we'll write a small piece of ML to transfer this memory back when the program executes. Assemble the following code at $4000:

```
4000 LDY #$0F
4002 LDA $4010,Y
4005 STA $0800,Y
4008 DEY
4009 BPL $4002
400B JMP $0900
```

8. Now we need to put the BASIC statement 10 SYS 16384 at the beginning of the program. This will jump to our transfer routine above ($4000=16384) and allow us to execute the program with a RUN command. Type M 0800 080F and enter the following bytes:

```
:0800 00 0D 08 0A 00 9E 20 31
:0808 36 33 38 34 00 00 00
```

9. Insert the COPY disk and save the program with S "COPY-Q",08,0801,4020
10. That's it - you're done. The program can now be LOADed and RUN like a BASIC program.

Thanks (and a $50 check) to Alan O'Dale from way Down Under for the following:

The program is BLITZ! BASIC COMPILER, Copyright 1983 by Prolic, Inc.

TYPE OF PROTECTION: The disk contains many errors but the program ,only checks for an error 22 on track 35, sector 4.

HOW TO COPY: Use any copy program which can copy error 22's.

HOW TO MAKE A WORKING COPY WITHOUT ERRORS ON THE DISK:

1. Load in a monitor at $8000 or above, such as HIMON.
2. Insert the original disk and load the compiler program with L "COMPILER",08
3. Change location $4243 from $42 to $40.
4. Save the program to a pre-formatted disk with S "BLITZ!",08,0801,4E9C.
5. That's it - you're done!

About all we can say for this is - it works! The modified program can be put on any disk; it contains the complete compiler. By the way, the original BLITZ! disk contains many programs on it besides the compiler. The other programs have all been scratched, but many can be recovered by 'unscratching' them and then file copying them from the disk (see PPM Vol. I for information about unscratching files). The extra programs appear to be demo programs for some old plotter model. Who knows what other buried treasure lurks on your commercial disks?

PROGRAM PROTECTION MANUAL FOR THE C-64 VOLUME II

Not just a third edition — a new and up-to-date manual covering the latest advances in program protection. Over 275 pages of valuable information. Topics include:

ENCRYPTION AND DECRYPTION

UNDOCUMENTED OPCODES

GCR RECORDING

CUSTOM DOS ROUTINES INCLUDING:
Halftracks
Extra Tracks and Sectors
Non Standard Sectors
Nybble Counting
Track Arcing
Density Changes
Synchronized Tracks

CARTRIDGES AND EPROMS
Using an Eprom Programmer
Cartridge Protection Schemes
Eprom Design and Manufacture

PROTECTING YOUR OWN SOFTWARE

TIPS FROM EXPERTS ON PROTECTION AND BACKUP

AND MUCH, MUCH MORE . . .

A disk is included to help explain the concepts

$34.95 + Shipping

* * * * * * * * * * * *

SINGLE SLOT EXPANSION BOARD
- FULLY SWITCHED, including GAME, EXROM, ENABLE AND POWER SWITCHES.
- LED indicators for type of cartridge
- RESET BUTTON
- COMPACT

$24.95 + Shipping

PROGRAM PROTECTION MANUAL FOR THE C-64 VOLUME 1
$29.95 + Shipping

1541 DISK DRIVE ALIGNMENT PROGRAM - VERSION 2

No special tools or knowledge are required to properly align the 1541 disk drive with this program (although average mechanical ability is required).

A WELL TESTED, PROVEN PRODUCT. NOW EVEN MORE ACCURATE! VERSION 2 FEATURES A NEW CALIBRATION DISK, SPECIALLY PREPARED ON SOPHISTICATED EQUIPMENT FOR EXTRA PRECISION.

NO FANCY SCREEN DISPLAYS OR HI-RES GRAPHICS. JUST A GOOD, SOLID DISK DRIVE ALIGNMENT PROGRAM — THOUSANDS ALREADY IN USE! SEE THE REVIEW IN THE OCTOBER 1984 ISSUE OF COMPUTE!'s GAZETTE.

$44.95 + Shipping

updates available to prior purchasers for
$20.00 + Shipping

* * * * *

PROGRAM PROTECTION NEWSLETTER

We will examine 3-5 (or more) programs each month. The protection schemes will be examined and broken down for aid in making archival copies. This will be a How and Why approach to program protection with special monthly articles covering the DOS operating system, product reviews and more.

$35.00 post paid in U.S. & Canada
$45.00 First Class post paid Foreign

* * * * * * * * * * * *

CSM
SOFTWARE, INC.

VISA AND MASTERCARD ACCEPTED
DEALER INQUIRIES INVITED
Shipping $[illegible] per item and [illegible] Foreign orders extra

CARTRIDGE BACKER
THE ORIGINAL CARTRIDGE BACKUP SYSTEM, NOW EVEN BETTER.

PACKAGE INCLUDES
1. EXPANSION BOARD, PROGRAM DISK AND USER'S MANUAL
2. CARTRIDGE BACKER software to backup 99% of the most popular cartridges to disk
3. ARTS, ACTIVE AND PARKER BACKER software to backup many disk programs
4. SUPERBOOTER — to create your own autoboots
5. SUPERSAVER — copies files to or from disk or tape
6. SOLD FOR ARCHIVAL USE ONLY!!

Cartridges are saved to disk with an autoboot and will run as-is. Unlike all other cartridge backup systems, *NO EXTERNAL RAM BOARDS, ROMS OR CARTRIDGES ARE REQUIRED.*

FOR THE C-64
$54.95 + Shipping

* * * * * * * * * * * *

NUMERIC KEYPAD
FOR VIC and C-64
NO SOFTWARE REQUIRED
100% COMPATIBLE WITH ALL SOFTWARE
MORE FUNCTIONS THAN KEYPADS COSTING $79.95

$64.95 + Shipping

PRODUCT REVIEWS

SEPIC - by STARPOINT SOFTWARE

ell, as you all know, Bruce Hammond (of Starpoint Softeware) promised us a ersion of the ISEPIC product to review (I did give him my credit card number, o it's not like we're asking for it for free). He assured me that I would ave a working version of this program for review in our newsletter and that e would be receive it soon - OVER TWO MONTHS AGO!! Well, after a number of hone calls to Starpoint Software (by the way, Bruce was never 'available' to nswer my calls) we have still not received our ISEPIC.

ortunately we have a really great bunch of readers who were willing to help s out with product reviews. Here is a brief consensus of what our readers ell us:

. A dealer tell us about **'problems with the hardware (circuit board), otherwise the program is pretty good'**
. A user tells us that **'all in all he is disappointed and wishes he would have saved his money'**
. A user reports **'ISEPIC failed to crack almost 40% of the programs that he tried it on'**
. Finally a programming consultant from Oregon reports **'that he was disapointed in ISEPIC because it is not what its advertising makes it out to be - yet he still would be willing to buy it again. He also wished that they would improve the product so that it was more effective.'**

nere you have some comments from our readers. Nobody was enthusiastic about ne program, most were disappointed with it. Only one said they would buy it gain, if given the chance to do it over.... I just wonder if our readers omments have anything to do with why we have not received our copy of ISEPIC et????

---

e have heard of another product called 'CAPTURE' that appears to do just what EPIC claims to do. The main difference between these two products is the ict that CAPTURE allows the user to put the program on EPROMs. Think of it, eing able to stop virtually any program, at any point and save it onto a artridge. Have you ever thought about putting your favorite word processor or ta base on cartridge and having that program boot up instantly.....
e **WILL** be getting a copy of CAPTURE for review next month.

ASTHACK'EM by BASEMENT BOYS SOFTWARE

YOU'RE WANTING TO SPEND $30.00 AND FEEL LIKE YOU REALLY GOT YOUR MONEY'S)RTH - **THIS IS A PRODUCT THAT YOU'LL WANT TO BUY!!!** This thing is great!! seconds to copy a disk with two drives. Less that two minutes for a single rive back up. 68 seconds for the MSD, and it does protected software. Plus e program has already been updated to copy many more protected disks. The py-protected disks are backed up via a template (special set of commands) at may be tailored to most any set of parameters or protection scheme. nis program was written by Mike Henry (formerly of DISECTOR fame).

---

ore product reviews will be in up coming issues of the newsletter. If you see program (good or bad) and think it would be of interest to our readers ease drop us a line and tell us your thoughts. If the program is great we ll let our readers know. If it's a turkey, maybe your recommendations will ve other readers some grief.

you hear of a new or exciting product just wait a few weeks before you end your hard earned money on it. We'll try to review each and every one.